2019

Optify – Next level optimization

BONUS PROJECT FOR CNG 350 SOFTWARE ARCHITECTURE

ORKHAN SALAHOV, RUSTAM QULIYEV, RASUL ALIYEV

LOGMYLOC | METU NCC

Table of Contents

Table Of Figures

1. INTRODUCTION

This is the requirements document for our graduation project for the next semester. It encompasses all the things that we are planning to accomplish and the way how we are planning to accomplish them.

1.1. ABSTRACT

The document starts by giving a brief overview about the project and very high-level idea as to what we are planning to accomplish. It goes through notations, and tentative requirements that the user should know about. Then it covers User requirements, System requirements and System specifications. These parts include all sorts of diagrams that are specified in the appendix part of the document in greater detail. Then it follows to architectural design of the software include class diagrams.

1.2. OVERVIEW

1.2.1. Description

LogMyLoc is a unique planning & networking tool that organizes your day using cutting edge algorithms so that you never feel the need to look at your schedule anymore. It offers you the most optimal place and time to do by analyzing your previous log of actions. Moreover, it offers a unique social network for you and your friends to meetup via analyzing the schedules and preferences of you all. Not to mention that you are the trainer of your own, personal scheduler, whom we call Jarvis, so you have full versatility over it.

1.2.2. Why LogMyLoc ?

LogmyLoc offers a unique alternative for our common scheduling methods that will make you entertained, save your time and never let you miss a single a deadline. It will help one in all aspects of their live by providing the best action to take at the moment whether it is where to eat, with whom to meet or what to study.

1.2.3. Scope & End User

The application will be best suit for people living in busy environment, where they have lots of things going on around them which is relatively hard to manage. Hence, our end-user will be a fresh start businessman living in a megapolis and having a busy social life. He needs a good assistant for him to manage his work, social life, personal and mental health at the same time since he is tired of managing it on in his own.

1.2.4. Goals & Objectives

Here are the goals and objectives of our software solution.

- Provide a user-friendly interaction to import data to the application
- Provide a mean of gamification to keep the user entertained while using the app
- Provide an interactive map of users around
- Provide strong and responsive algorithms for analyzing huge chunks of data
- Provide a means of categorizing entities in the application based on priority

1.2.5. Capabilities

The initial idea encompasses the following:

- Interactive map showing the real-time location of your friends
- Trainable virtual assistant (Jarvis)
- Means to view your histogram and schedule
- Way to import data to the app and communicate with Jarvis via voice
- Way to import data to the app and communicate with Jarvis via text
- Add friends and create groups
- Interact with people/groups and organize your day

1.2.6. Future Evolution

Here are initial ideas for expansion:

- Unsupervised training based on surrounding sound and conversation
- Future predictions based on previous logs
- Sematic analysis of the task and its importunateness using NLP

1.3. TENTATIVE NON-FUNCTIONAL REQUIRMENTS

Here is the list of vital non-technical requirements that we have come up tentatively.

1.3.1. Usability

There should be an intuitive way of communication with the software. UI/UX design should be carried out thoroughly, so the users don't feel alienated and lost upon downloading the app. As mentioned, 2 ways of communicating with the software will exist: typing via text and voice recognition. The user should select either mode in the settings of the app. Jarvis should support interactive communication and feedback to users' responses.

1.3.2. Internalization

The software should exist on multiple languages and should account for one's culture, religion and worldview while providing solutions. Jarvis should also support different languages and genders.

1.3.3. Data Integrity

In Terms & Conditions for the application it will be mentioned that we collect all the data of your without distributing it to a third-party. Since the data we gather will comprise one's location that may include his/her workplace, place of living, gym etc. we may not allow that data to be easily cracked. Because of that we more secure architecture will be chosen later.

1.3.4. Reliability

The users need to be sure that the software is, indeed, providing the best possible alternative upon choosing the next action(s) to be undertaken. For that reason the decision algorithms will be greatly refined till perfection. Also Jarvis will self-learn once it's suggestion has been overlooked and internally it will assign priorities for each individual task to be accomplished.

1.3.5. Performance

Although not that crucial, however, an average user response shouldn't take more than 2 seconds to be completed. Example scenario: A group of 5 people want to assign a time to meet for 15 minutes today or tomorrow and Jarvis should come up with 3 best alternative time slots for that to realize. A query of such degree should take 2 seconds at most to be completed.

1.3.6. Maintainability

Since we have quite some goals for the future of the project, we need to lay the foundation so that the maintenance and evolution won't take too much iterations. For that reason we are SOLID designs should be, mostly, followed.

2. PROJECT MANAGEMENT

2.1. INTRODUCTION

This chapter mostly covers our plan in organizing the software development process for the future in order to effectively manage, develop, deploy and test our project within the timing constraints of 2 academic semesters. It includes some information about us, the design patterns that we're going to implement, our objective and scope and risks related to project.

2.2. SCOPE

Our goal is to complete documentation, development, testing and deployment of the initial requirements abstractly covered in Part 1.2.5 within the time period of 2 semesters.

2.3. TEAM ORGANIZATION

Our team comprises 3 developers from Middle East Technical University Northern Cyprus Campus Computer Engineering Department.

2.3.1. Rustam Quliyev

Software Engineering student from Middle East Technical University NCC

2.3.2. Rasul Aliyev

Software Engineering student from Middle East Technical University NCC

2.3.3. Orkhan Salahov

Software Engineering student from Middle East Technical University

2.4. RISK ANALYSIS

What risks we might face during the development of the software?

2.4.1. Delayed project due to bad collaboration

Plan for risk reduction: We are going to use AGILE philosophy in developing the application by conducting daily session for our goals and divide up the project into different milestones. Also we are planning to use different progress tracking software like Slack to get a good performance metric of our development.

2.4.2. Change of requirements

Plan for risk reduction: As we have mentioned previous we are going to use SOLID principles while creating the classes so that maintainability is easier. Also, we will plan out our requirements according to priority and deliver with increments.

3. USER REQUIRMENTS

3.1. DESCRIPTION

This section will include the user requirements of our software. Those requirements are created for people with non-technical background to be able to understand our platform.

3.2. REQUIREMENTS

1. The application should have a usable, intuitive Graphical User Interface(GUI) with vivid, inspirational colors similar to that of Material Design.
2. If the user is signing up as a new user, it should provide a brief, explanatory, animated tutorial which can, optionally, be skipped.
3. Application should prompt the users to sign in or log in before starting to use the application. Only signed users can use the application. Log in can be done using Google Plus and Facebook as well.
4. There should be a clear Terms & Policy in the app.
5. At the first use of application the user should be, optionally, prompted to enter his/her schedule to the app, and edit his/her profile. It can be edited later anytime through the app's corresponding section.
6. The entries of the schedule must have a name, time, type, priority, frequency, privacy and notification settings and do not disturb regime. Location of the entry should be optional but advisable to enter as well.
7. Upon the first time using the do not disturb regime the app should make the user to pick the apps for which the notifications will be blocked. This can be altered anytime from the settings menu.
8. Types are associated with a color. There are some predefined types by the app itself but the user can add types and reassign colors to them in the settings menu.
9. One can view the entries settings by clicking on it. Further it can edit or delete the entry as well.
10. After each new entry the priorities of all entries will be standardized to fit a certain distribution.

11. The entries of the schedule of the user should be displayed as a colormap according to their priority or by the colors of their type. Default setting being the typed colors. User can switch between these views.

12. There should be a part to view statistics about your schedule. Which are attendance rate for each entry, pie chart on the percentage of time one spends on each type of activity.

13. HOW THE ATTENDANCE IS CALCULATED

14. There should be a colorful map, on the main screen, showing the current place, the place of the next destination according to your schedule, the time where you need to be at that entry, the time it will take to get there, a calculated best route, other visible friends around you.

15. The map should be able to change its color schema if the weather darkens.

16. By clicking on a person icon on the map one can view his entries, which are marked as public and have locations attached to them, throughout the day. Also one can view her entire public schedule and compare with ones' own one. One can also create an event with that person. If the person is not on your friend list on can easily send a friend request as well

17. The users can create events thus becoming the <u>administrator</u> of the event. Events can be either fixed time or <u>live</u> time (non-fixed). If the time is live the apps decision system will try to find the best slot for it in your schedule according to the priorities of the entries and the settings of the event.

18. Any event hosted by the user can be shared in 2 ways: within the app or with a sharing link. A person who has been anyhow invited to an event and accepted it will have the status of <u>tracking</u> the event.

19. The administrator of the event can see the list of participants and edit it, tracking list and adjust settings regarding the event and the permissions and priorities of the participants.

20. For the live mode as new people are accepting the invitation the heatmap of best timeslots will be dynamically altered on the events schedule menu which is available to admin and all the privileged users. The admin will also see the top 5 possible timeslots as a heatmap within the time interval provided.

21. Admin of an event time can place the timeslot anywhere in the heatmap mentioned in *18*.

22. The admin of a live event can fix and unfix the timeslot. Upon fixing the timeslot the members will receive a message to add the event as an entry in their timeslot with the decided time.

23. The app will be able to able to detect new entries for the schedule by reading the emails, phone calls and etc. They are controlled by user in the privacy settings.

24. There should be notification feature in the app.

25. User can have different <u>activity status</u> within the app which are: online, offline, invisible.

26. User will have a simple profile to include his/her name, photo, description, status and home address.

27. There should be an external API to control the schedule and maintain the events. The API should be integrated with popular voice assistants like Siri and Alexa.

28. There should be a slide in the main menu.

29. Setting menu shall include the privacy settings

30. The friends can be organized into groups.

31. Admin can add/remove people from the participants list. As well as remove group.

32. All members of the group can see the group Profile, members list and group schedule.

4. SYSTEM REQUIREMENTS

4.1. DESCRIPTION

System requirements are a more detailed, lower level of the user requirements of Section 3. The numbers are numbered as subsection of those provided in User Requirements to facilitate the traceability.

4.2. REQUIREMENTS

1.1. The graphical user interface of the app should be as designed as in Section XXX

2.1. The tutorial shall cover how to add entries to your schedule

2.2. The tutorial shall cover how to use the map

2.3. The tutorial shall cover how to use the event functionality of the app

2.4. The tutorial shall cover the privacy settings of the app

2.5. Tutorial can be skipped if wished so

3.1. User must provide username, password and email in order to register.

3.2. An email and the username should be assigned to only one account.

3.3. During registration the user can provide his/her Name, surname, profile photo, description and home address.

3.4. An email with the validation link should be sent in order to validate the account.

3.5. Upon login the user can enter either email or username with his password.

3.6. The app should support registration through Facebook and Google Plus.

3.7. As the last step of registration the user should be prompted to modify his/her schedule.

3.8. While registering through Facebook or Google Plus the app should provide the list of ones friends that are using the app and a button to add them to your friends in the app. This step can be skipped if wished so.

4.1. Terms & Policy should be signed by user during the registration process. Users who have not signed the Terms & Policy cannot proceed with their registration.

4.2. Terms & Policy should clearly state that the data gathered from the users will be stored in our servers, however, in no way be distributed to a third-party organization.

5.1. At the first us the user should get the animated tutorial.

5.2. After the animated tutorial the user will be brought to the main menu

6.1. An entry for the schedule shall have a name. The name of the entries should not be unique.

6.2. An entry for the schedule shall have one or many starting and ending times, in other words for each entry multiple blocks with starting and ending times may exist.

6.3. Based on previous entry records the ending time should be calculated after setting the starting time of a block of time in the entry.

6.4. An entry for the schedule shall have a type. The predefined types are: Default, Academic, Sport, Recreational, Personal, Social, Work.

6.5. An entry for the schedule shall have a certain priority number which is a float between 0 and 100.

6.6. An entry for the schedule shall have a frequency, which is one of the following: single-time, every day, once a week, once a fortnight, once a month, once a season, once a year. Default one should be the single-time.

6.7. If the user selects any frequency except single-time one, he/she will be prompted to enter the ending time of the entry. Without setting the ending time user wont be able to proceed adding the entry to the schedule.

6.8. An entry for the schedule shall have a privacy settings, which are one of the following: Public, Friends, Private, Friends except…, Specific friends…, Groups …, Custom. Default being Friends.

6.9. An entry for the schedule shall have notification settings. The notification settings should select the number of minutes/hours/days before which to notify the user about the event. Default notification should be with in-app notification 30 minutes before the entry.

6.10. The user can select means of notification which can be in-app notification, mail or SMS.

6.11. The location can be set for the schedule as well. There should be a button which will display the map to select the location by hand or can type in the details of the address to the requested field.

6.12. An entry for the schedule shall have a do not disturb checkbox which won't allow any external notifications to pop-up during that time.

7.1. Upon the first use of the do not disturb mode the user should be directed to the settings for this mode, where she/he will allow the LogMyLoc to block the notifications from the selected apps.

7.2. This settings will be available from the settings menu as well.

8.1. The types for the entity are associated with special colors. The types can be added by adding a type name and associated color in the settings menu. The colors can be set using the color templates provided or through a detailed colored palette.

8.2. Events that have been added to user's schedule will have a specific event flag and will be distinguished in the UI.

9.1. One can view the entries for schedule my clicking on it. All the details of the entry can be changed anytime be the owner of that entry.

10.1. After adding each entry to the schedule the entries

11.1. The heatmap will be based on the priority of the entries. The 0 priority will be green while the 100 priority will be red and anything in between will be a relative mix of these two colors. Here is the sample of the gradient.

Figure 1 | Heatmap Color Schema

11.2. The type of the color should be based on the type of the entry.

11.3. There should be a checkbox to toggle between the two schedule viewing options.

12.1. There should be a button to enter the statistics viewing mode of the schedule.

12.2. There should be a general statistical variable showing the global attendance rate.

12.3. There should be a pie chart display the time one spends on each type of activity.

13.1. Adsasd

14.1. There should be a map on the main screen. Main screen meaning being the place where users have the most interaction.

14.2. The map should display my home (if provided in) with a special home icon.

14.3. The map should display me with a green bubble in the map inside of which is my profile photo.

14.4. The map should display the user with gray bubble if the user in invisible mode.

14.5. The map should display my friends as blue bubble in the map inside of which is their profile photo.

14.6. Stranger (non-friends) should be displayed as white bubbles in the map

14.7. The map should display my next entry as a rectangle in the map. The color of the rectangle should be the color of the associated type of the entry.

14.8. There should be a start time associated with the next entry displayed on cliking the rectangle as well.

14.9. If the entry doesn't have a location associated with it, the rectangle should be placed on top of my person bubble in the map.

15.1 The map will change its color scheme to black if the time is 8pm and switch back to white when time is 6am.

16.1. Upon clicking an holding a user bubble on the map new sub bubbles with specific icons appear on top of that user, each representing some function.

16.2 Apart from that on the map users entries will be placed on the map.

16.3. If the user is not on your friends list the first bubble will be to add that person to your friend list by sending a friends request.

16.4. One bubble will be to view that person's profile

16.5. One bubble will be to view that person schedule and compare to your own one.

16.6. One bubble shall be to add that person to a group

16.7. One bubble will be for sending an invitation for an event to that person.

16.8. Only those entries for which you have privilege to view can be seen in the schedule.

16.9. Upon clicking on ones' own bubble all the events with locations shall appear in the map and sub-bubbles should appear to choose either online or invisible mode

17.1. There should be a button for a user to host an event thus becoming the administrator of the event.

17.2. The administrator of the event shall provide the event name, event type, the location of the event (optional), description of the event(optional), cover photo of the event(optional), timing mode selector.

17.3. The event type can be one of the entry types that the user has

17.4. There can be 2 types of timing mode: Fixed time, Live time.

17.5. If live time is selected the preferred starting and ending date interval as well as preferred starting and ending time interval is entered. Also the user shall enter the event duration.

17.6. If fixed time is selected the exact start and ending time is entered.

18.1. In-app sharing allows you to invite anyone visible to you to the event. It supports a search feature based on username, Name or Surname.

18.2. There should be a clipboard for the admin and privileged users to copy the link of the event. Of course anyone can send the link for the event to anyone through various ways.

18.3. Those who have proceeded through the invitation link will be sent to the event page where he/she can see all the details of the event and where a button exists to track the event.

18.4. Those who opted to track the event will be put, by default, to the tracking list.

18.5. Admin can accept or decline the user from the tracking list.

18.6. The system shouldn't allow same user reapplying to the same event more that 2 times.

19.1. Admin can view the participants list from the events menu. From that menu he/she can assign privileges and priorities to each participant. Admin is also included in participants list.

19.2. Participants list should be filtered by the priorities and closeness.

19.3. Privileges can be any combination of the following abilities: invite people, view schedule, accept/discard people, see participants list, see the event link, kick users, assign priorities, assign privileges, edit group description and profile picture.

19.4. Priority can be a float between 0 and 1 default staying at 0.5. The priority will be a hyperparameter for the scheduling algorithm determining the weight of each user.

19.5. Admin and privileged users can view the tracking list of the incoming requests and either accept of discard them. Tracking list should be filtered by friends first.

19.6. Admin can adjust event settings which are the following: update description, profile photo, time range, member capacity (default unlimited), link visibility (default false), waiting list (default true), event type (Public, Private), participants can add members (default false), participants can see the participants list (default true),notification settings, schedule privacy (Anyone, Participants, None default being Anyone), Cancel event.

19.7. Upon cancellation participants will be notified and those who have had the event added to their schedule will have it removed.

20.1. There will be an event schedule section for the admin and all the privileged users.

20.2. The event schedule will be a heatmap of best possible time slots according to members schedules. With best time slot being green and worst one being red, and anything in between being a mix of those colors.

20.3. The schedule should give 5 best ranges for the time slots, for the person viewing the schedule.

20.4. The schedule should group consequent time-slots as a single range of time-slots.

21.1. The admin can place the event anywhere he wishes in the event schedule

22.1. The admin of live event can fix the timeslot after selecting one.

22.2. Upon fixing the timeslot the users will get notification to add the event to their own schedule.

22.3. The notification should provide a good way of comparing the schedule without the entry and with the entry inside the schedule.

22.4. Once the event has been fixed, also includes the fixed time events, the admin can unfix it only twice. If unfixed the users who had the event added to their schedule will have the event removed from it.

23.1. The app can read emails provided in settings menu and deduce to add new entry to the schedule.

23.2. If it happens so, the user will get notified to add the event into his/her schedule

23.3. Similar the app can listen to phone calls trying to deduce a new entry to be added.

24.1. There should be a notifications section in the app.

24.2. App should support push-in notification outside the app.

24.3. There should be different types of notification by clicking on which different actions are executed.

24.4. There is a notification to evaluate a friends request where the user can either accept or decline the offer or one can click and be directed to friends list where he can see all the friends.

24.5. There is a notification to accept the live event invitation. One can either accept or decline the invitation or can click and be directed to event’s main page.

24.6. There is a notification to add the fixed event invitation to schedule. One can either accept or decline the invitation or can click and be directed to event’s main page. Upon approval the schedule of the user will be updated

24.7. There should be a notification if someone has accepted your friend request.

24.8. There should be a notification if one has been accepted to the group

24.9. There should be a notification if event time has been fixed

24.10. There should be a notification if group has been created

24.11. There is a notification to accept group invitation.

25.1. Users can have 3 activity statuses: Online, Offline, Invisible.

25.2. If the user in using the app by default he/she is considered online.

25.3. The user can change his/her activity status to invisible and continue using the app whilst the people won't be able to see his/her presence in the app.

25.4. If the user exits the app in the database the user will be tagged as offline.

26.1. User's profile can be updated anytime through the profile section

26.2. User's profile includes a square profile photo

26.3. Upon changing the profile photo the user is provided with tool to crop his/her profile photo to square format

26.4. User's profile include a Name, Surname, Description and Home Address.

26.5. User's profile also displays ones status of visibility.

27.1. External API should support Siri and Alexa

27.2. API should support anything mentioned for the schedule. That includes: Create entry, Delete entry, Modify entry

27.3. API should support anything mentioned for the event. That includes: Create event, Delete event, Invite friend to event.

28.1. Slider should include my profile

28.2. Slider should include my friends list

28.3. Slider should include my group list

28.4. Slider should include my events

28.5. Slider should include Settings

29.1. Privacy settings shall include allowance to listen to the voicemails, default being 'not allowed.

29.2. Privacy settings shall include allowance to read the e-mails, default being 'not allowed'.

29.3. Privacy settings shall include the activity status of the user, default being ‘Friends. Other categories are as follows: Public, Friends, Friends except…, Specific friends…, Groups …,Custom, Invisible.

29.4. Privacy settings shall include the distance from the user from where strangers can see her, default being ‘5 meters’. This means that regardless of the activity status of the user except ‘Invisible’ the stranger within that range will be able to detect the users.

29.5. The privacy and schedule will be allowed to be changed anytime from the respective settings menu.

29.6. The privacy settings include the visibility of Friends, Groups, Locations in the profile. Each being one of those : Friends, Public, Private, , Friends except…, Specific friends…, Groups…, Custom….

30.1. Friends can be grouped.

30.2. Groups have name, profile photo, description.

30.3. Owner of the group can add new members

30.4. Owner of the group can remove a member

30.5. Owner of the group can remove the group totally.

31.1. There is a members list in the group.

31.2. There is a button to view Profile of the group

31.3. There is a button to view Schedule of the group.

31.4. The schedule of the group should have an add event button. Upon clicking on which the default event creation process executes.

31.5. The is Settings section in the group.

31.6. In the Settings menu the Owner can assign privileges to group members which can be: Event creator, Administrator, User.

4.3. USE CASE DIAGRAM

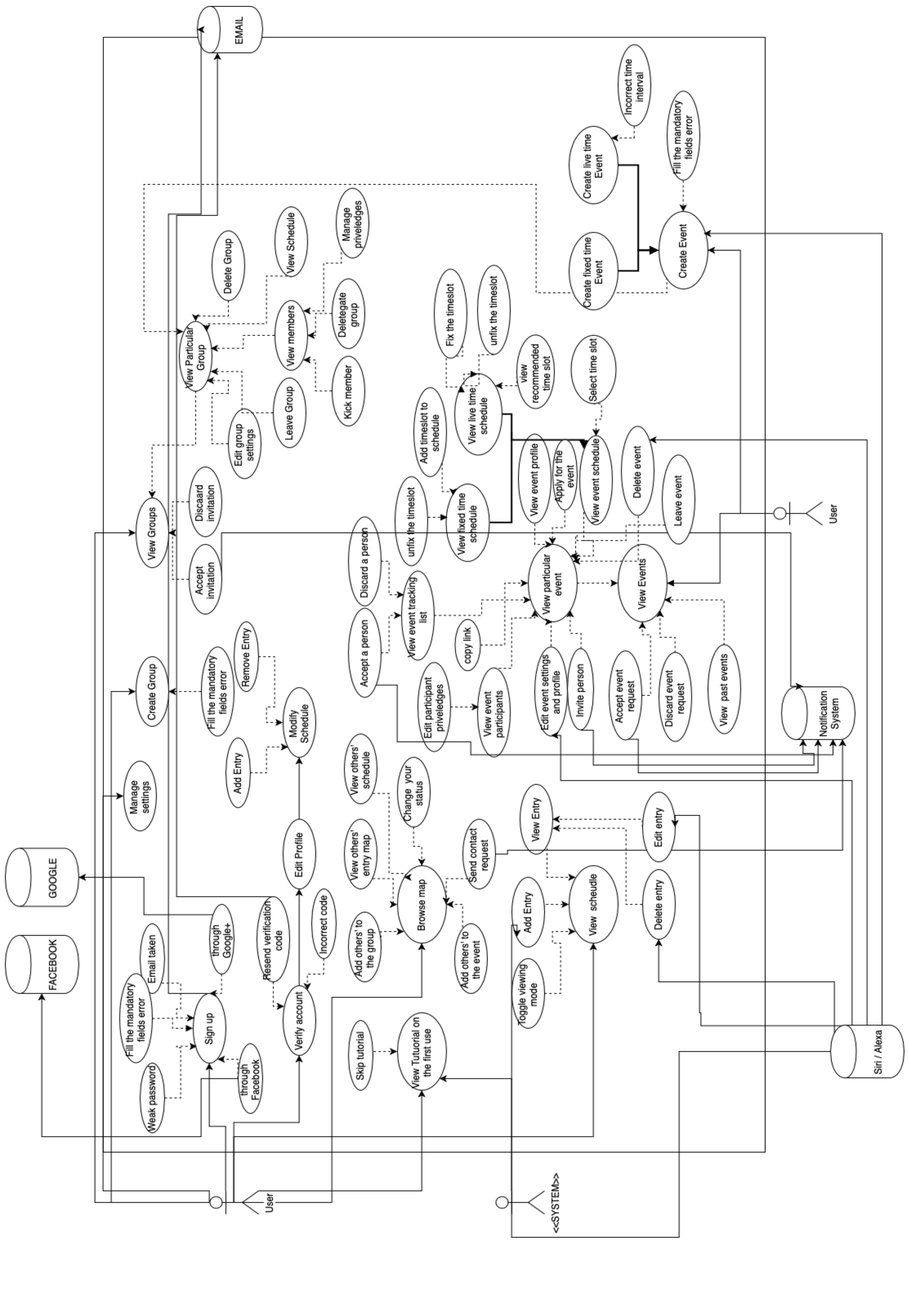


Figure 2 | Use Case Diagram

Continue …

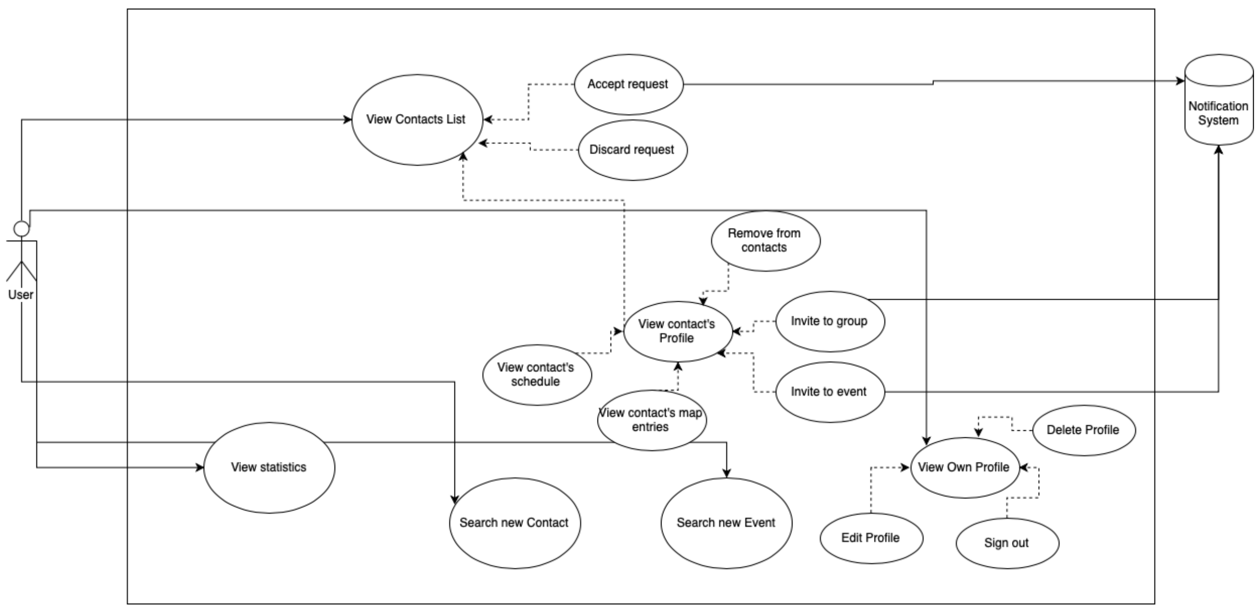


Figure 3 | Use Case Diagram Part 2

4.4. ARHCITECTURE DIAGRAM

4.4.1. Level-0 Context Diagram

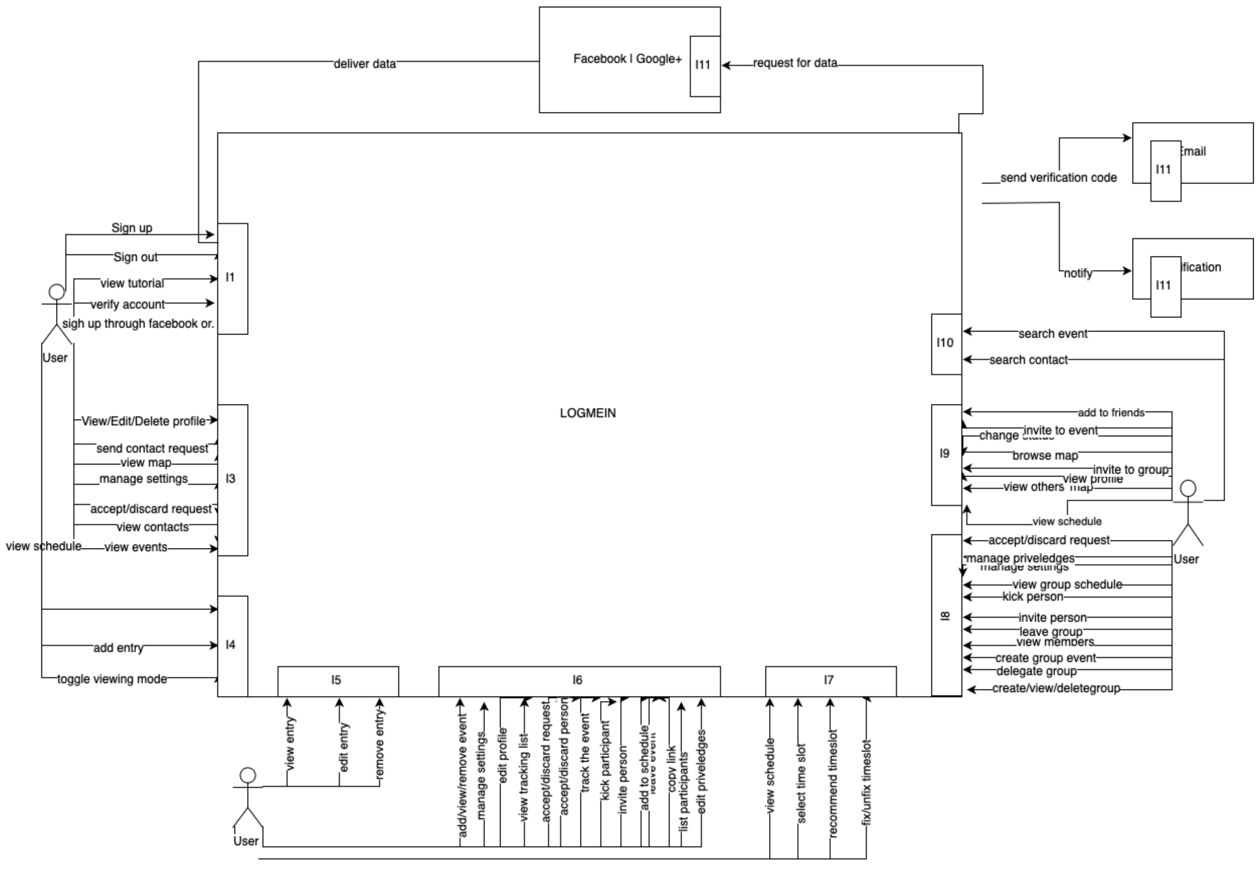


Figure 4 | Architecture Level 0

4.4.2. Level – 1 Architecture Diagram

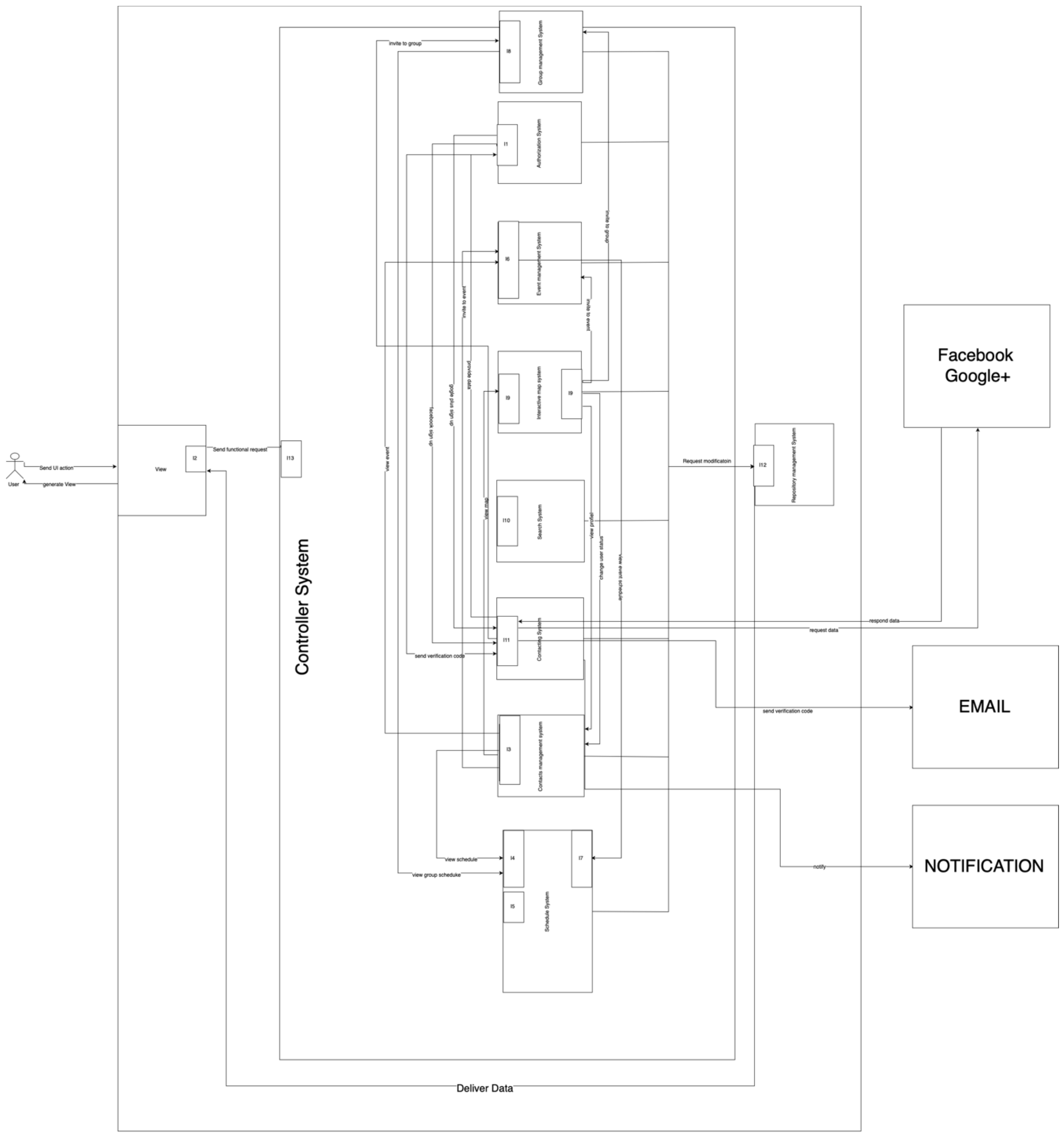


Figure 5 | Architecture Level – 1

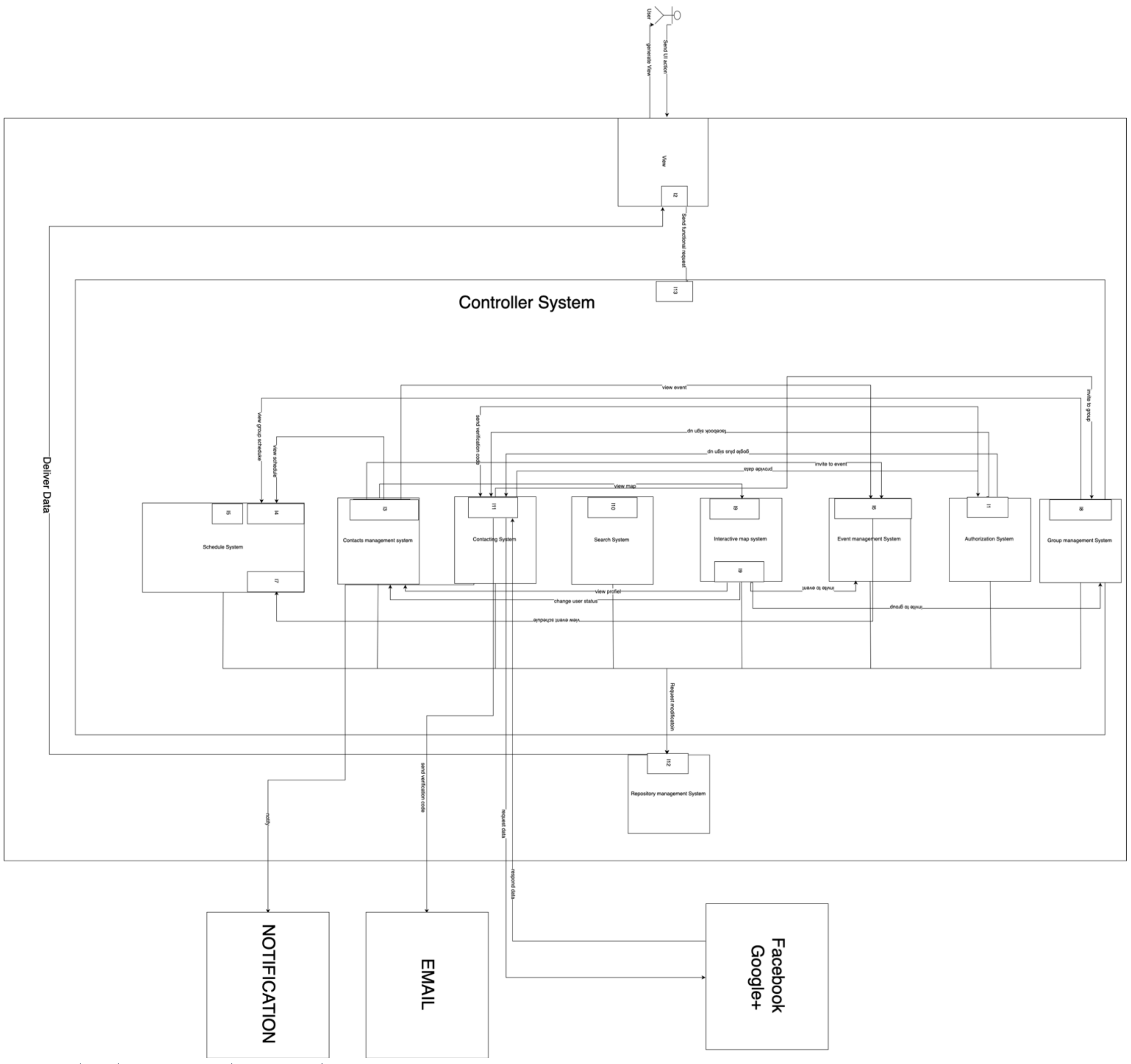


Figure 6 | Architecture Level - 1 rotated

For this application we have used MVC with Repository model. Also we have assumed that Notification is an external system on its own.

4.5. INTERFACE DEFINITIONS

4.5.1. <I1: < Authorization interface

```
Interface Authorizationinterface {
        Sign_up()
        Log_in()
        Sign_out()
        Verify_account()
        Facebook_sign_up()
        Google_p_sign_up()
        Sendverificationcode()
}
```

4.5.2. <I2: View Interface

```
Interface View {
        Listen_user_input()
        Generateview()
        acceptData()
}
```

4.5.3. <I3: User Profile Interface

```
Interface UserProfile {
        viewProfile()
        editProfile()
        deleteProfile()
        send_contact_request()
        view_schedule()
        view_event()
        view_events_map()
        change_activity_status()
        manage_settings()
        accept_request()
        discard_request()
}
```

4.5.4. <I4: User Schedule Interface

```
Interface UserSchedule {
        View_schedule()
        Add_entry()
        Toggle_view_mode()

}
```

4.5.5. <I5: Schedule Entry Interface

```
Interface ScheduleEntry{
        View_entry()
        Edit_entry()
        Remove_entry()
}
```

4.5.6. <I6: Event Interface

```
Interface Event {
        Add_event()
        Add_group_event()
        Accept_invitation()
        Discard_invitation()
        View_event()
        View_schedule()
        Remove_event()
        Manage_settings()
        View_participants()
        Copy_link()
        View_tracking_list()
        Accept_from_tracking_list()
        Kick_participant()
        Track_event()
        Edit_priveledges()
}
```

4.5.7. <I7: Event Schedule Interface

```
Interface EventSchedule {
        viewSchedule()
        select_timeslot()
        fix_timeslot()
        unfix_timeslot()
        recommend_timeslot()

}
```

4.5.8. <I8: Group Interface

```
Interface Group Schedule {
        Accept_invitation()
        Discard_invitation()
        Create_group()
        Remove_group()
        Delegate_group()
        View_group_schedule()
        Invite_person()
        Kick_person()
        Leave_group()
        View_group()
        Create_group_event()
        View_memebers()
}
```

4.5.9. <I9: Map Interface

```
Interface Map{
        Browse_map()
        Change_user_status()
        Invite_to_event()
        Invite_to_group()
        Send_contact_request()
        View_profile()
        View_entry_map()

}
```

4.5.10. <I10: Search Interface {

```
Interface Search {
        searchUser()
        searchEvent()
}
```

4.5.11. >I11: Contacting System {

```
Interface Contacting {
        sendEmail()
        sendNotification()
        contact_facebook()
        contact_googlep()
        deliverdata()
```

4.5.12. I12: Repository Managing Interface

```
Interface Repository {
        readQuery()
        writeQuery()
}
```

4.5.13. I13: Controller Interface

```
Interface Controller {
        Accept_user_request()
        Return_system_response()
}
```

4.6. CLASS DIAGRAM

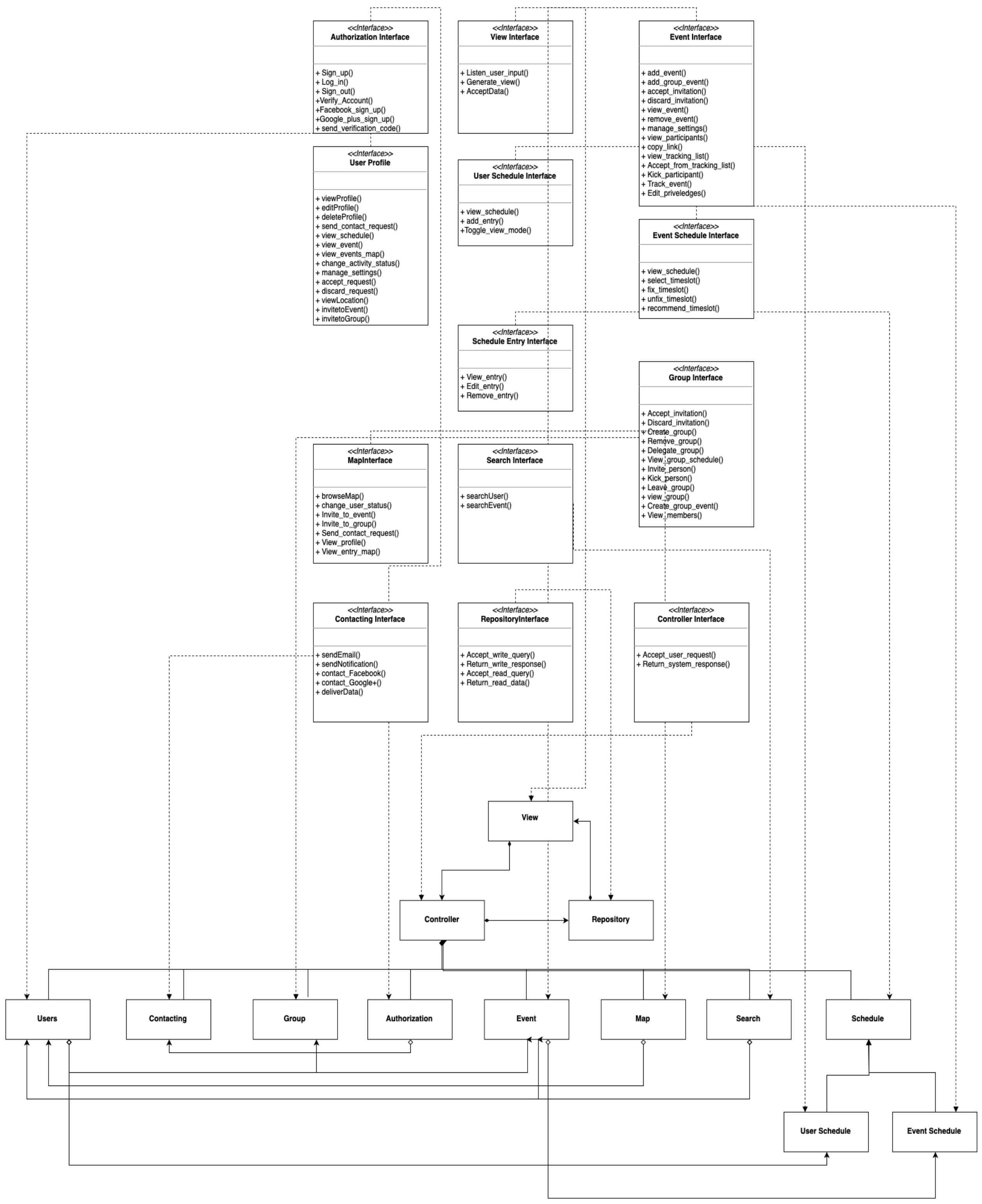


Figure 7 | Class Diagram

Because of restricted time we couldn't create Architecture Level – 2 and went straight to Class Diagrams

4.7. DFD DIAGRAM

Since our time is limited we have covered the DFD diagram for the view schedule use case and its extended use cases.

4.7.1. DFD Level – 0 (Context Diagram)

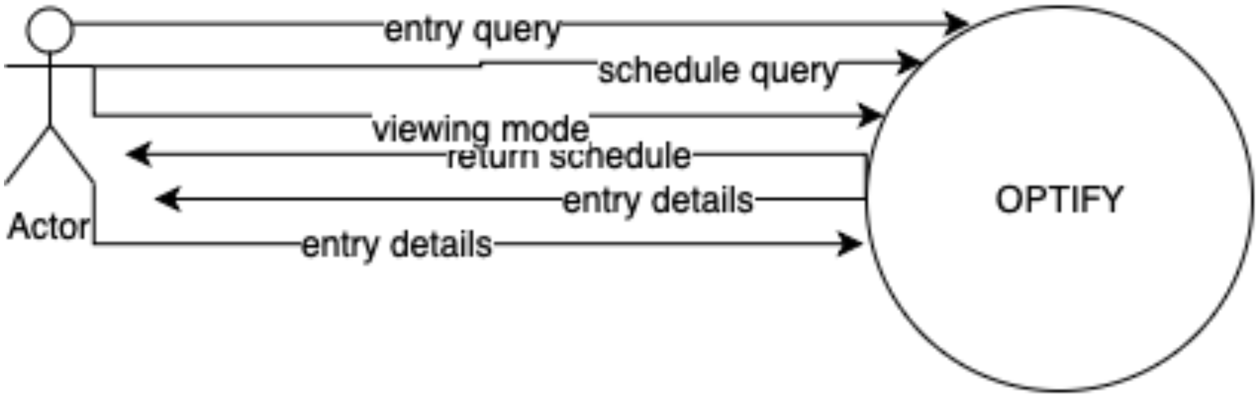


Figure 8 | DFD Level -0

4.7.2. DFD Level – 1

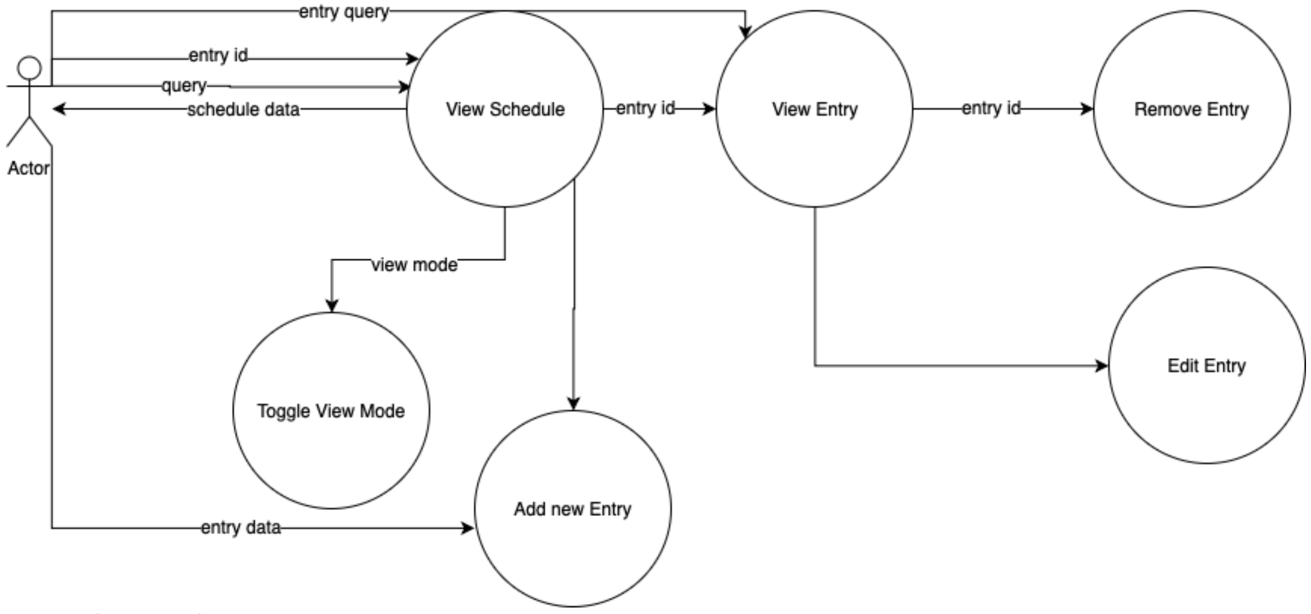


Figure 9 | DFD Level - 1

4.7.3. DFD Level – 2

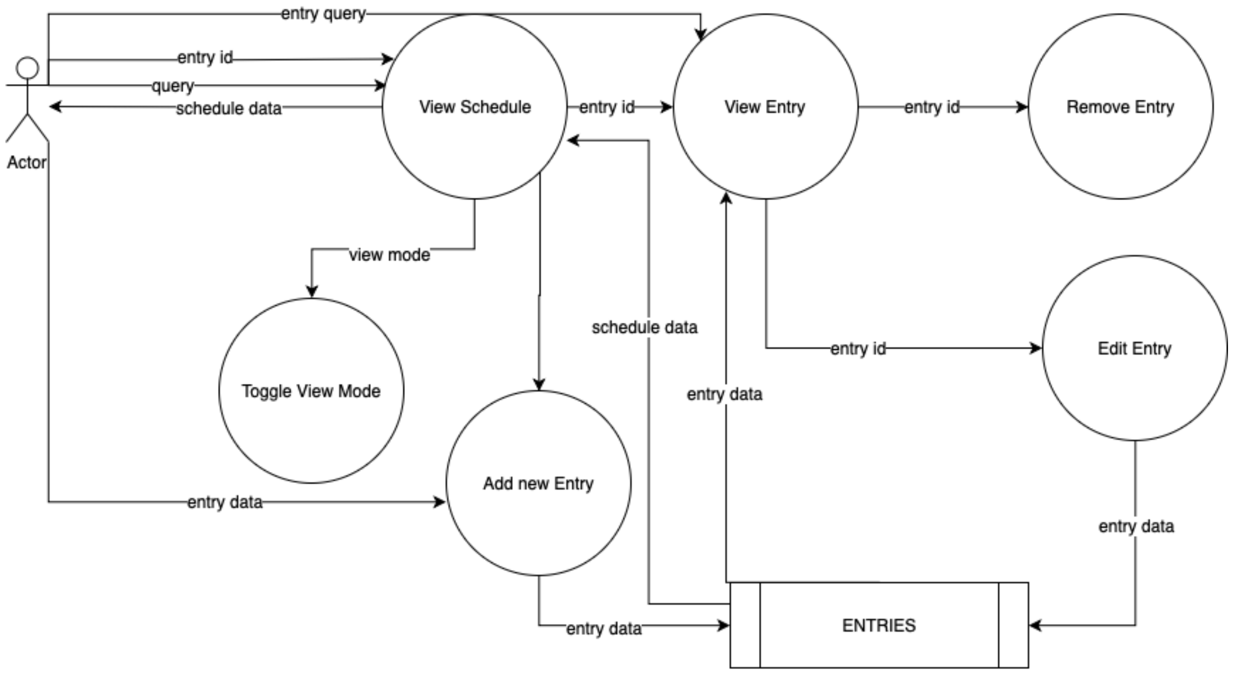


Figure 10 | DFD Level - 2

4.8. SEQUENCE DIAGRAM

In this section we will provide you with sequence diagram for Create Event Use Case.

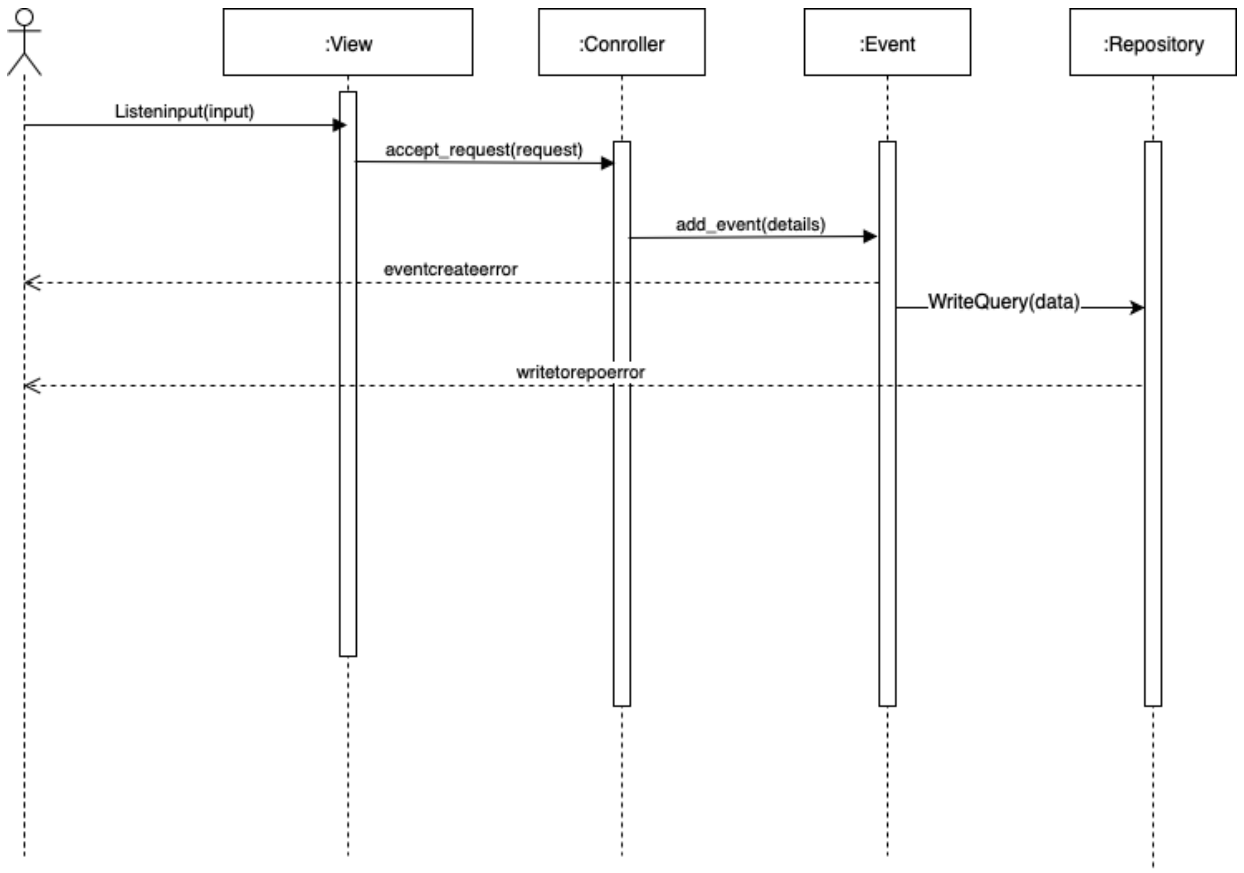


Figure 11 | Sequence Diagram

4.9. DATABASE DESIGN

Simple ERD Diagram

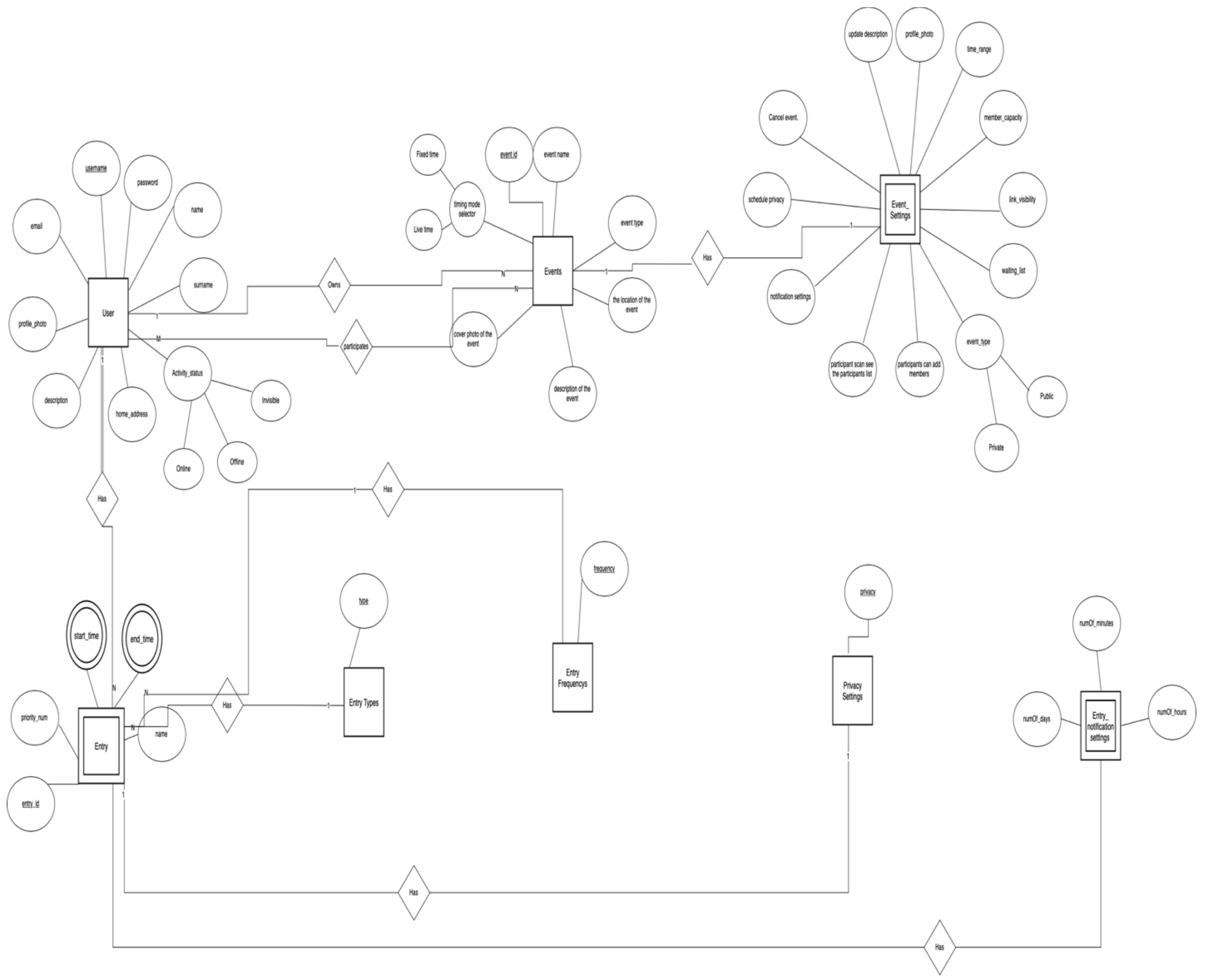


Figure 12 | Simple ERD Diagram

4.10. STATE DIAGRAM

4.10.1. Login State Diagram

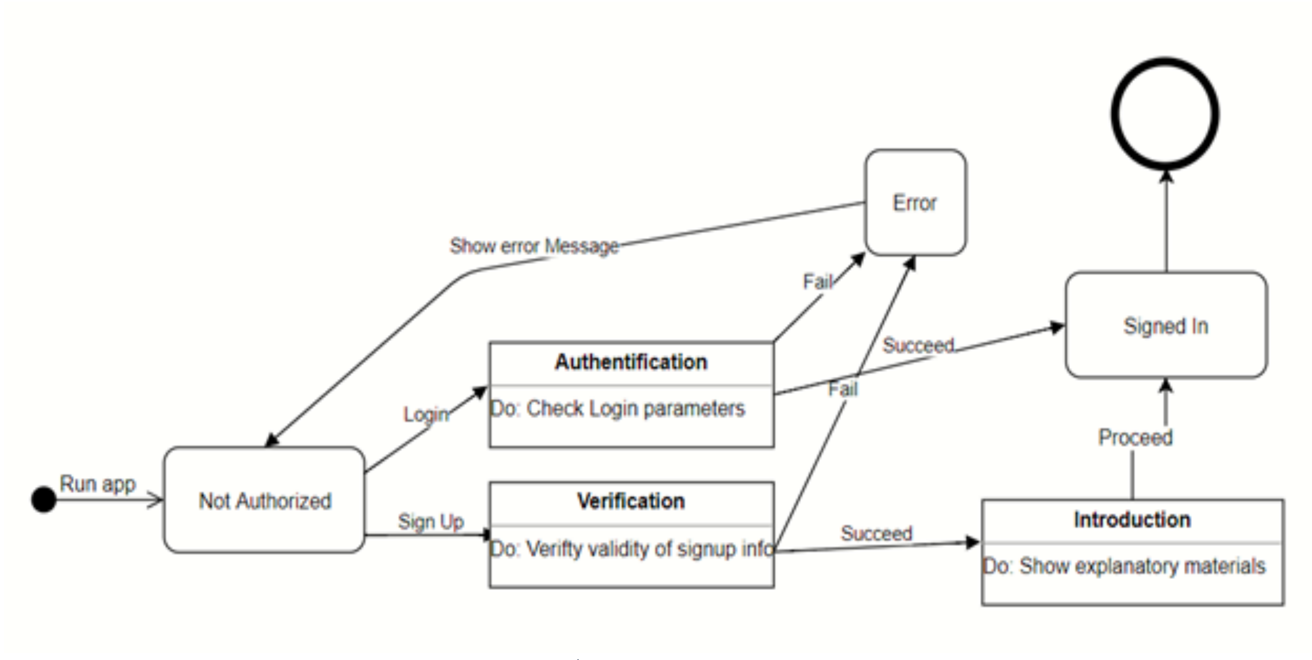


Figure 13 | Login State Diagram

4.10.2. Event State Diagram

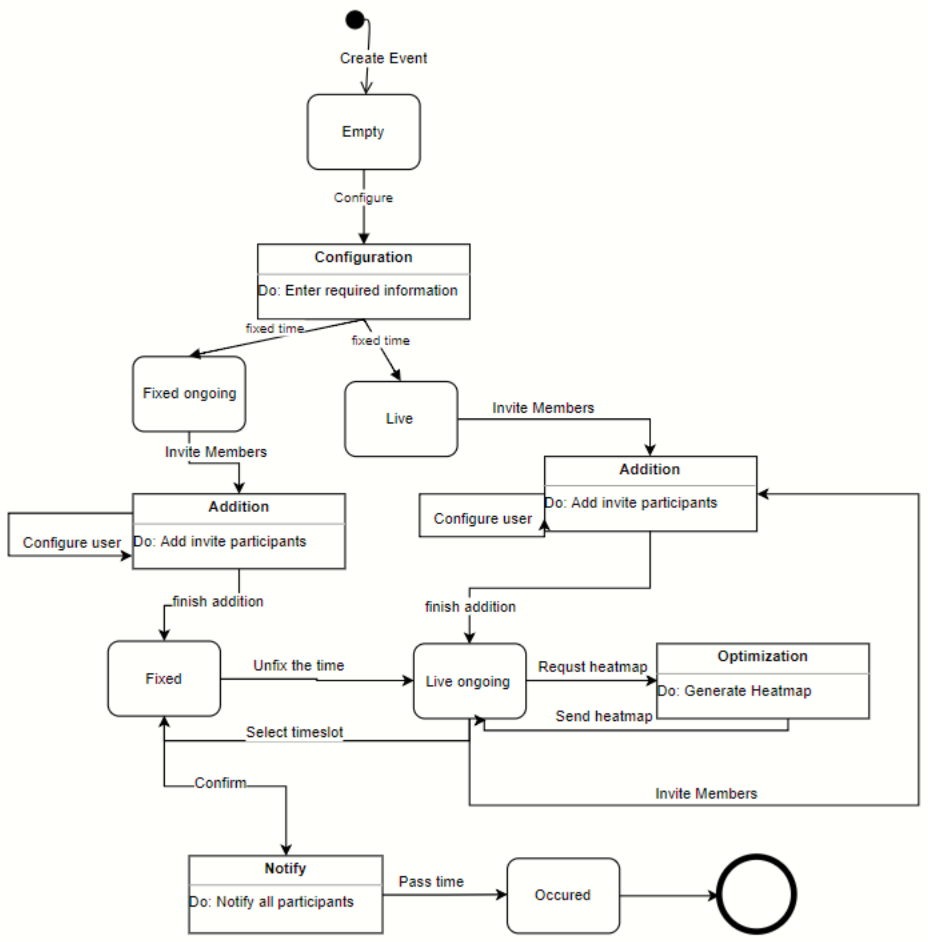


Figure 14 | Event State Diagram

4.10.3. Main State Diagram

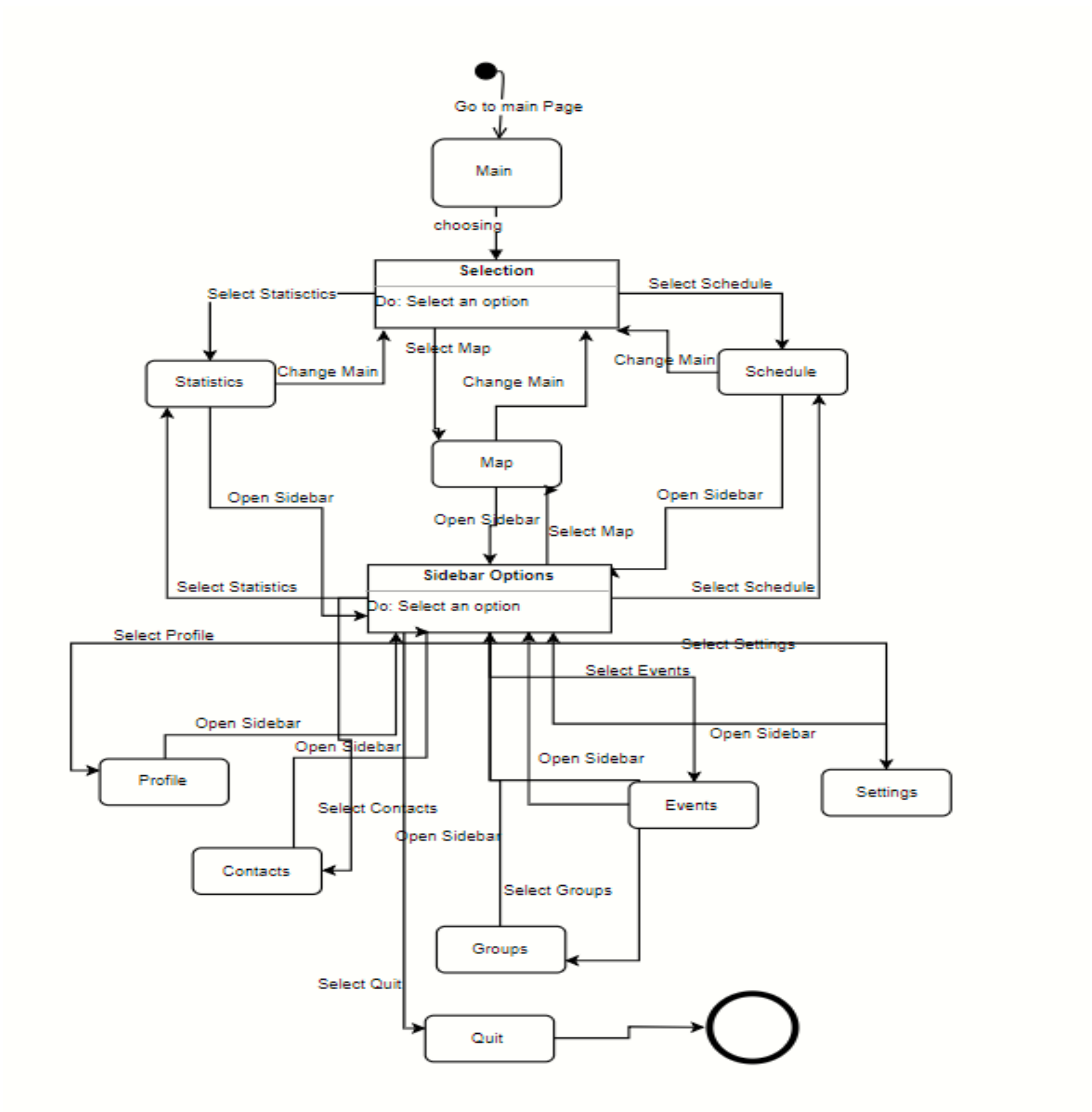


Figure 15 | Main State Diagram

5. GRAPHICAL USER INTERFACE

This section includes the snapshots of the application in terms of design.

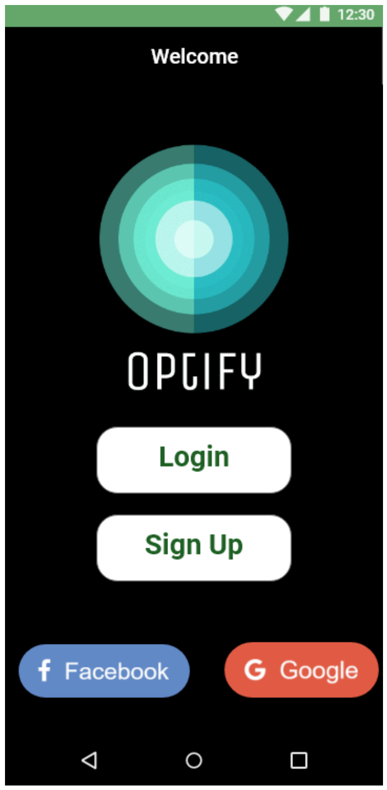


Figure 16 | GUI Welcome

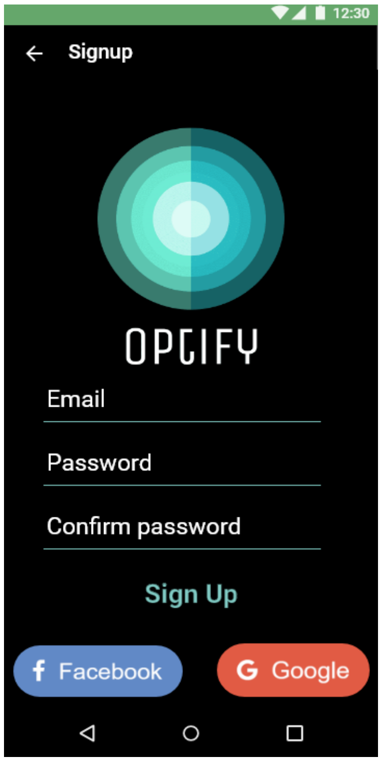


Figure 17 | GUI Sign up

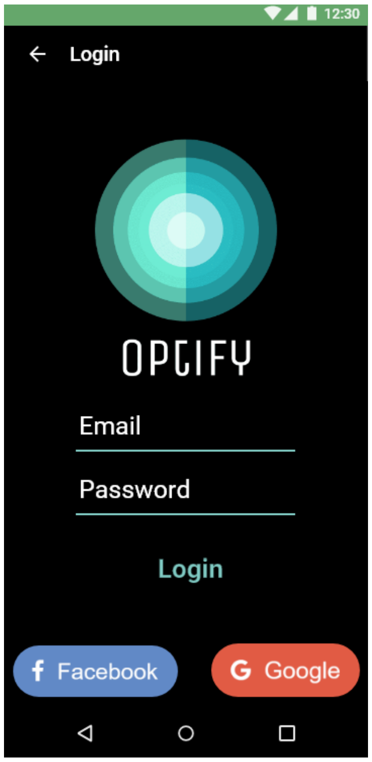
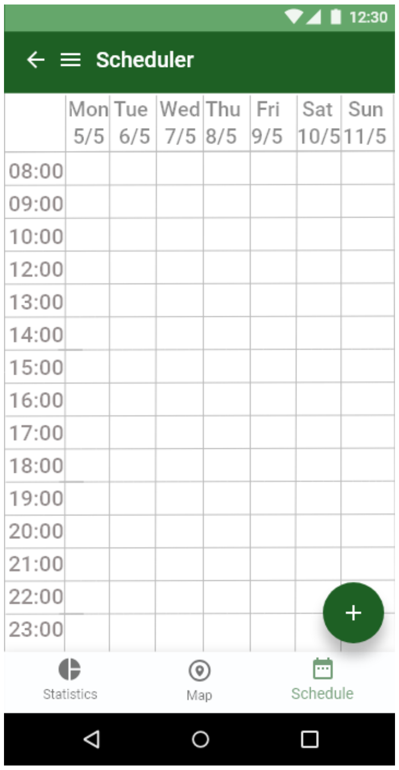
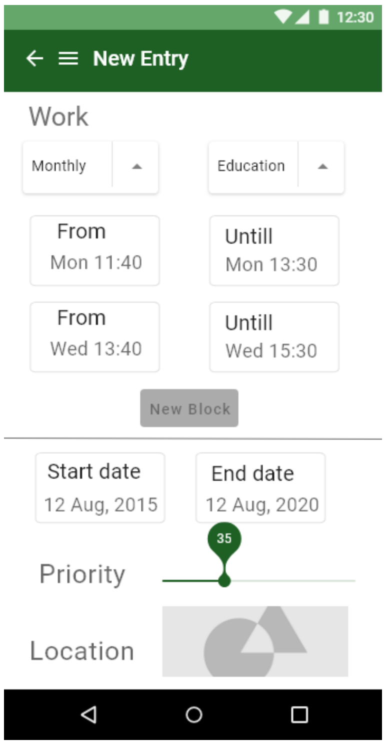


Figure 18 | GUI Sign In



	Mon 5/5	Tue 6/5	Wed 7/5	Thu 8/5	Fri 9/5	Sat 10/5	Sun 11/5
08:00							
09:00							
10:00							
12:00							
13:00							
14:00							
15:00							
16:00							
17:00							
18:00							
19:00							
20:00							
21:00							
22:00							
23:00							

Figure 19 | GUI Schedule



Work

Monthly

Education

From Mon 11:40

Untill Mon 13:30

From Wed 13:40

Untill Wed 15:30

Start date 12 Aug, 2015

End date 12 Aug, 2020

Priority 35

Location

Figure 20 | GUI Insert Entry

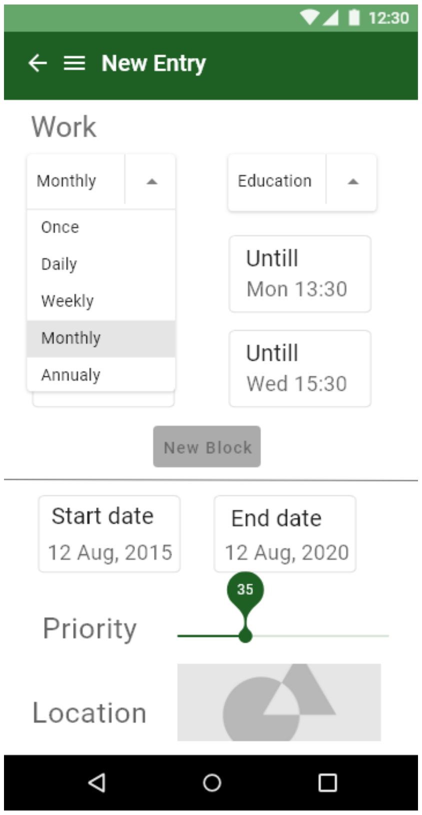


Figure 21 | GUI Insert Entry Frequency

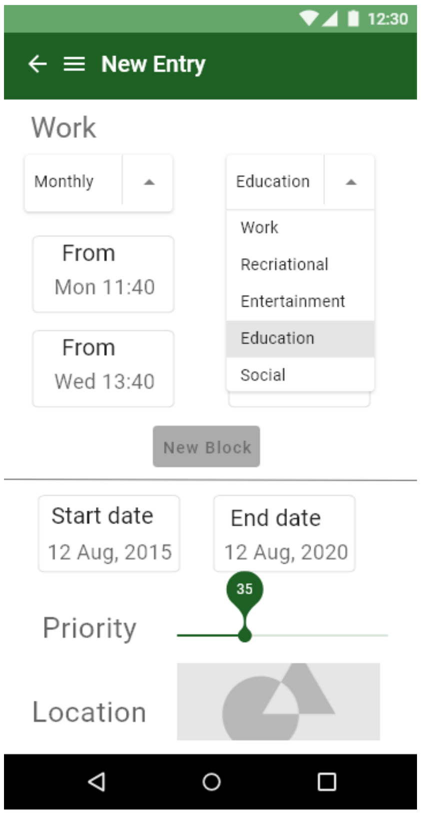


Figure 22 | GUI Insert Entry Types

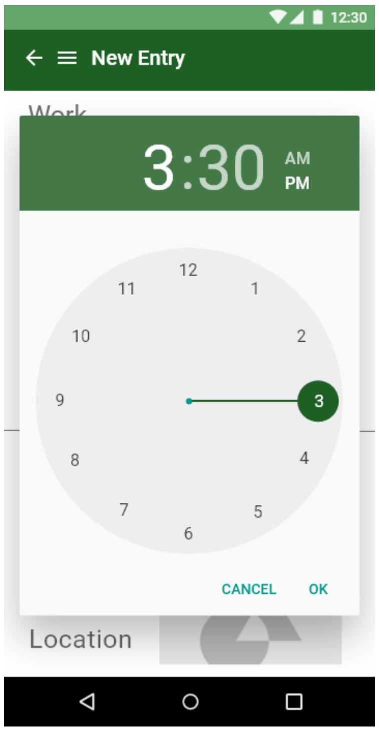


Figure 23 | Insert Entry Clock

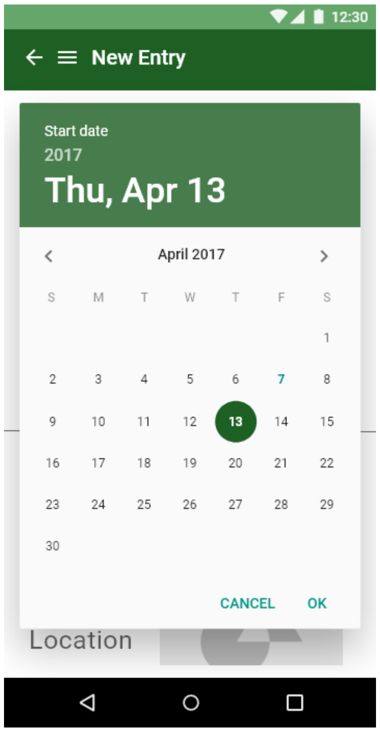


Figure 24 | GUI Insert Entry Calendar

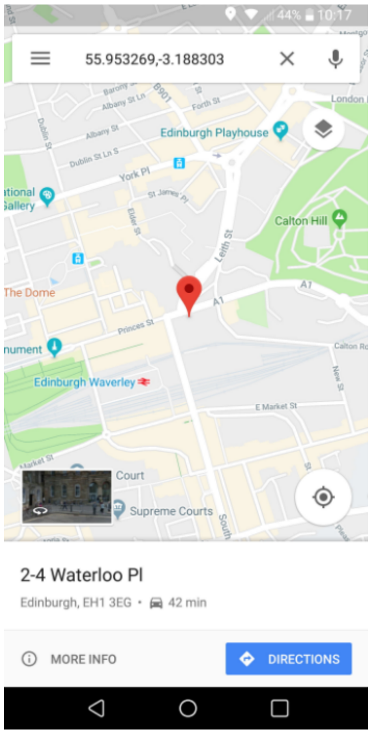


Figure 25 | GUI Insert Entry Map

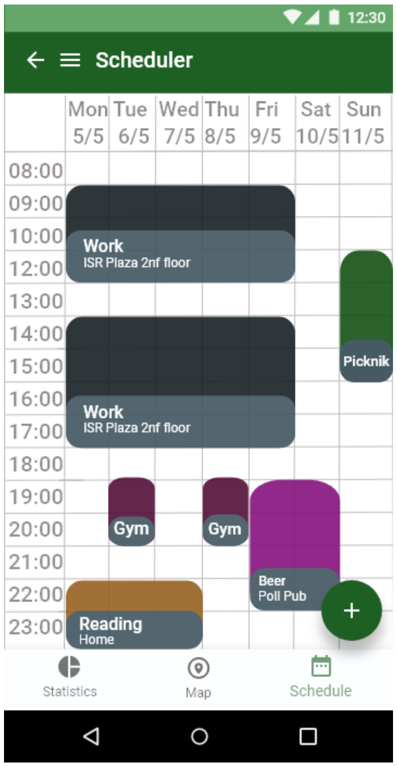


Figure 26 | GUI Schedule full

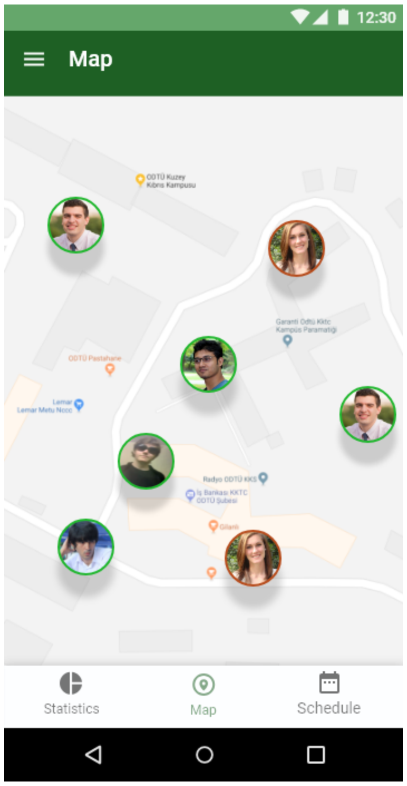


Figure 27 | GUI Map with Friends

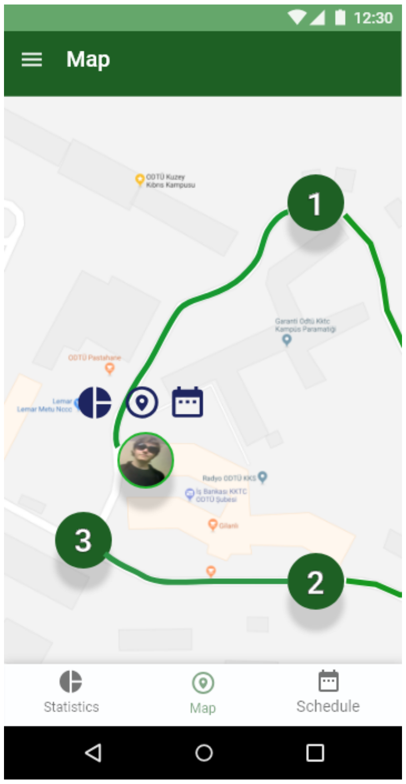


Figure 28 | GUI Map with destinations

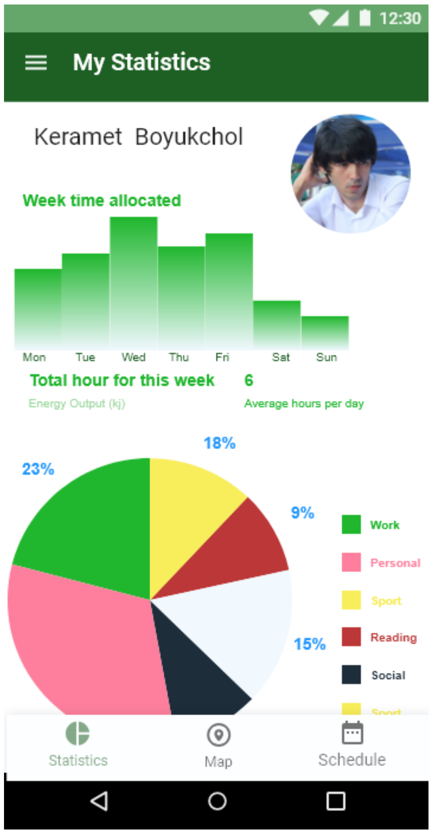


Figure 29 | GUI Statistics page

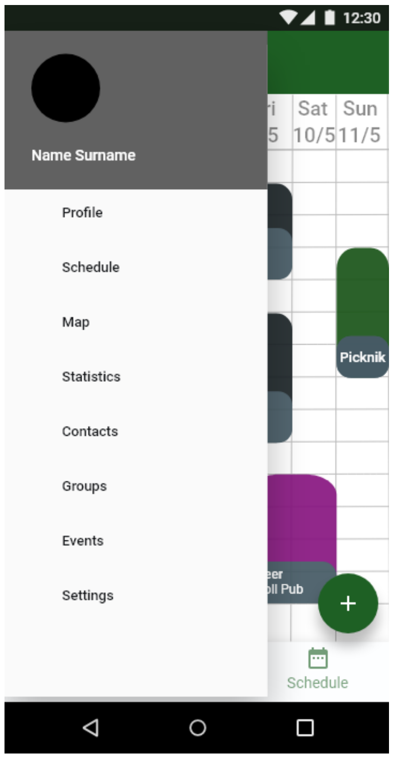


Figure 30 | GUI SideBar

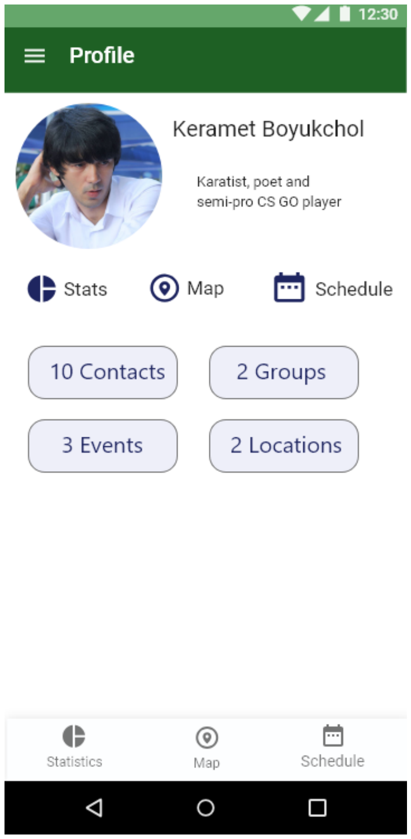


Figure 31 | GUI Profile

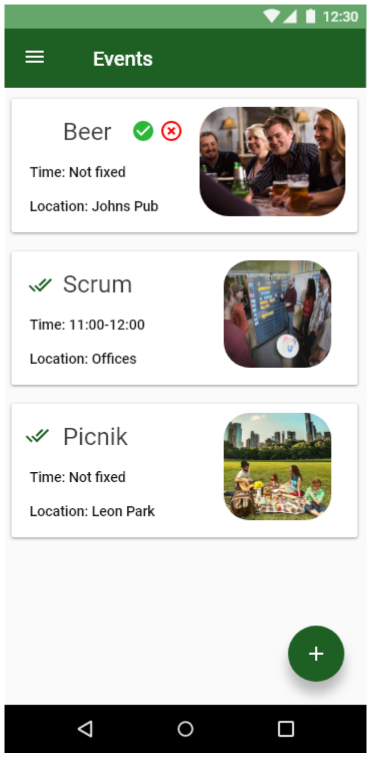
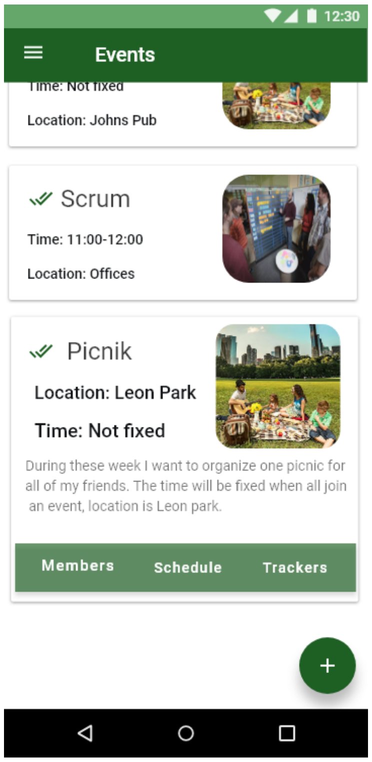


Figure 32 | GUI Events


12:30

Events

Time: Not fixed

Location: Johns Pub

Scrum

Time: 11:00-12:00

Location: Offices

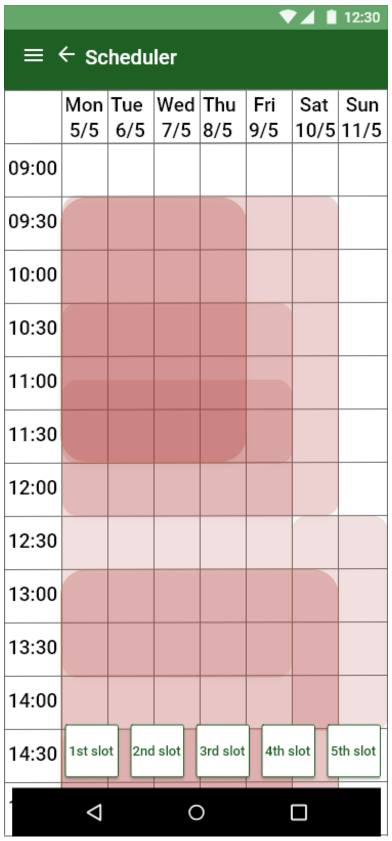
Picnik

Location: Leon Park

Time: Not fixed

During these week I want to organize one picnic for all of my friends. The time will be fixed when all join an event, location is Leon park.

Figure 33 | GUI Events 2


12:30

Scheduler

	Mon 5/5	Tue 6/5	Wed 7/5	Thu 8/5	Fri 9/5	Sat 10/5	Sun 11/5
09:00							
09:30							
10:00							
10:30							
11:00							
11:30							
12:00							
12:30							
13:00							
13:30							
14:00							
14:30							

Figure 34 | GUI Event Schedule

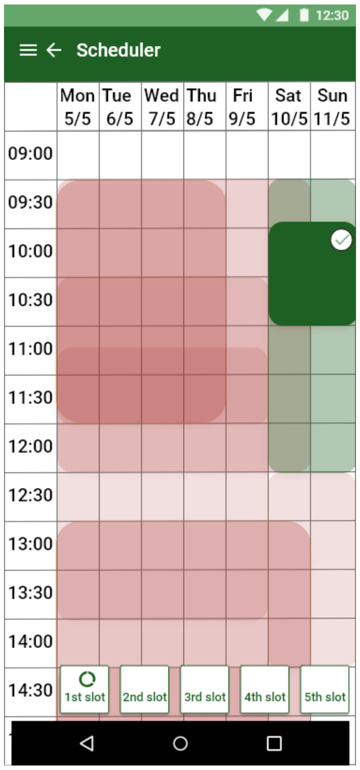


Figure 35 | GUI Heatmap 1

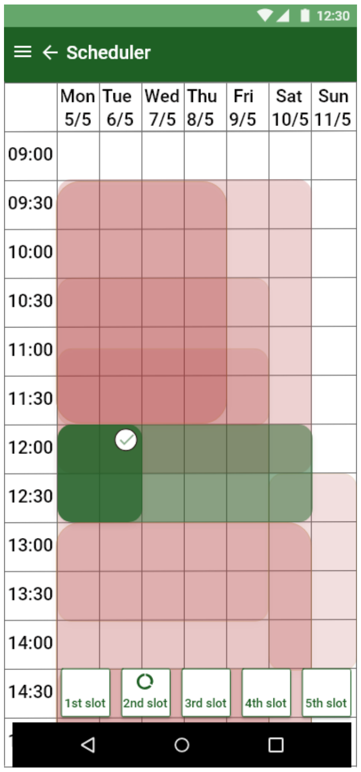


Figure 36 | GUI Heatmap 2

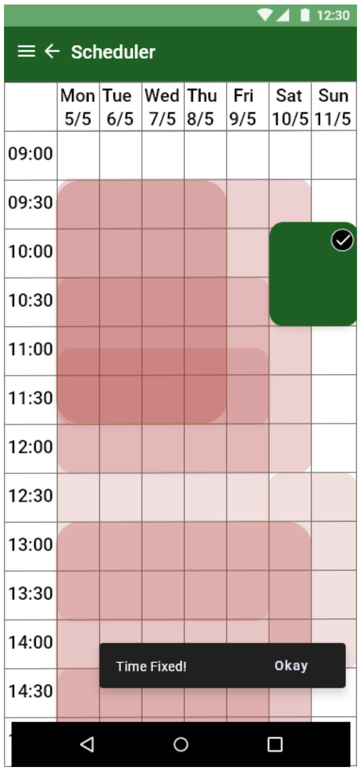


Figure 37 | GUI Heatmap 3

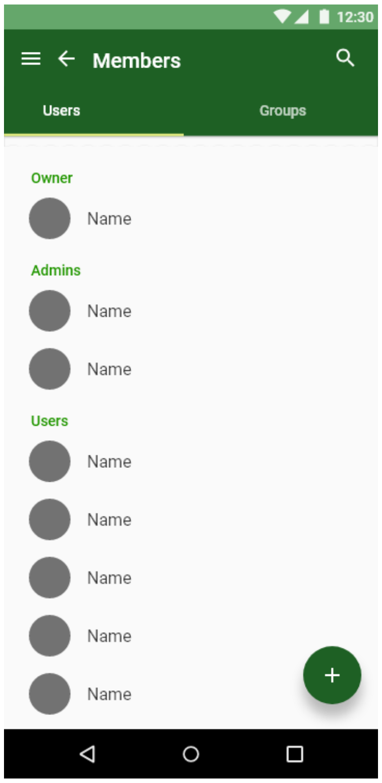


Figure 38 | GUI Event Members

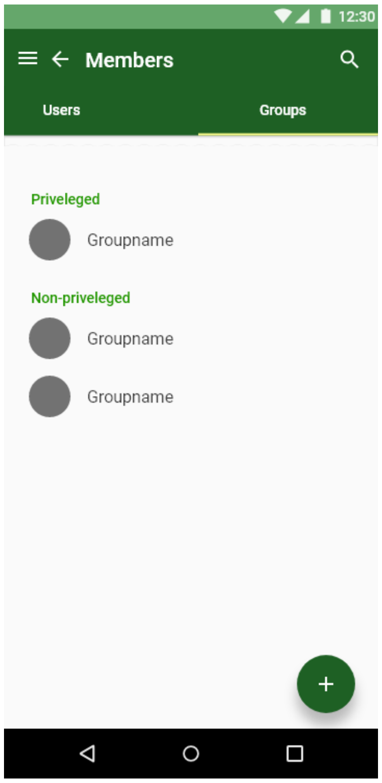


Figure 39 | GUI Event Groups

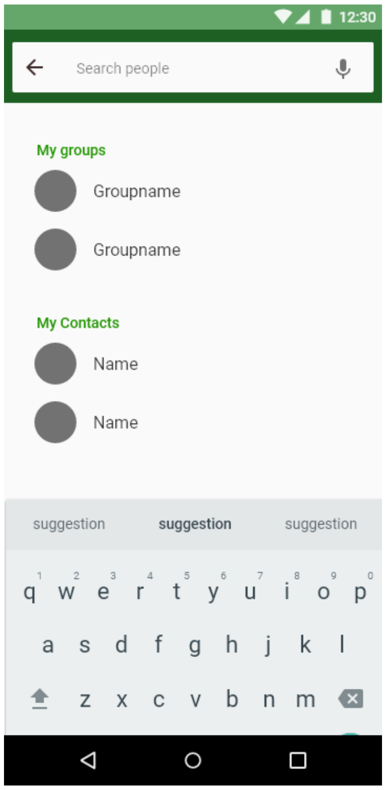


Figure 40 | GUI Event member Search

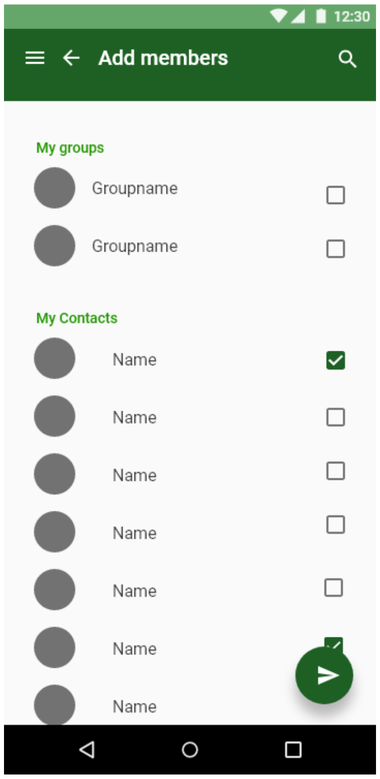


Figure 41 | GUI Event Member Invite

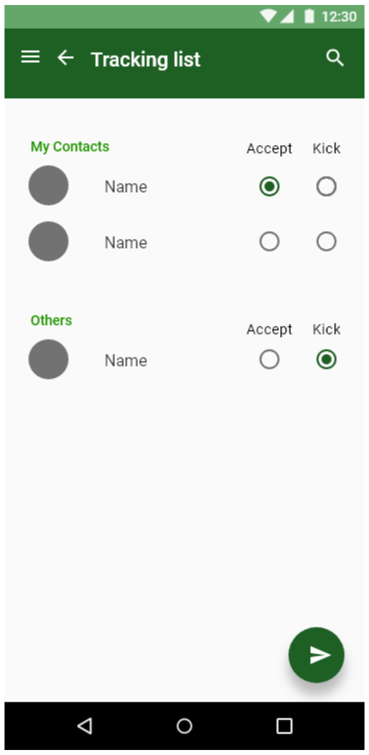
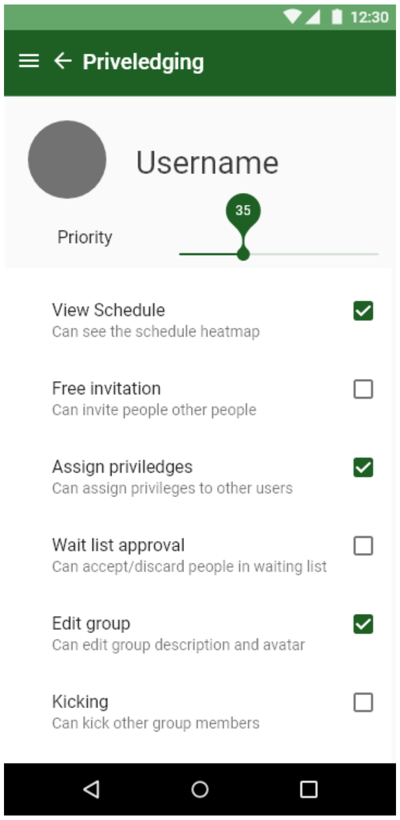


Figure 42 | GUI Event Tracking List


12:30

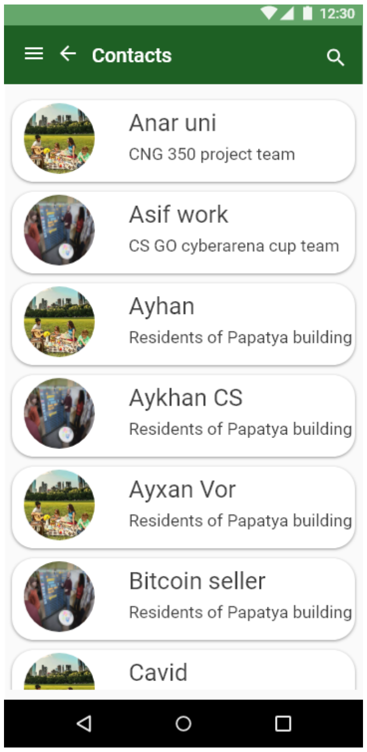
Priveledging

Username

Priority 35

View Schedule Can see the schedule heatmap	✓
Free invitation Can invite people other people	☐
Assign priviledges Can assign privileges to other users	✓
Wait list approval Can accept/discard people in waiting list	☐
Edit group Can edit group description and avatar	✓
Kicking Can kick other group members	☐

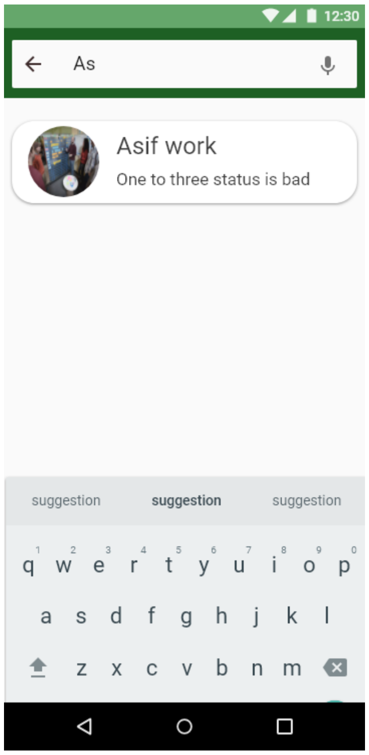
Figure 43 | GUI Event Privileging


12:30

Contacts

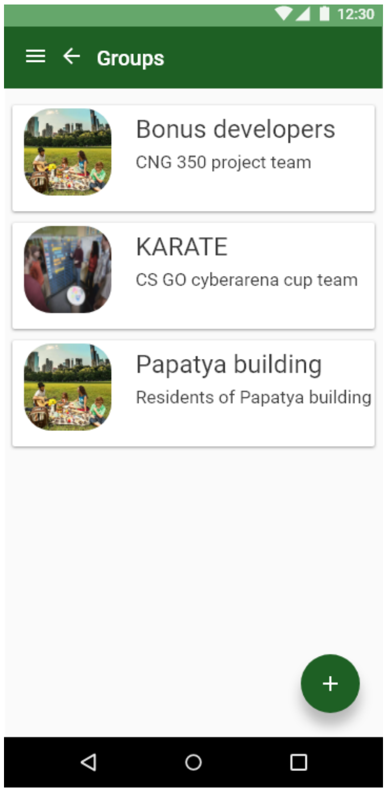
Anar uni	CNG 350 project team
Asif work	CS GO cyberarena cup team
Ayhan	Residents of Papatya building
Aykhan CS	Residents of Papatya building
Ayxan Vor	Residents of Papatya building
Bitcoin seller	Residents of Papatya building
Cavid	

Figure 44 | GUI My Contacts



Asif work
One to three status is bad

Figure 45 | GUI Contacts Search



Bonus developers
CNG 350 project team

KARATE
CS GO cyberarena cup team

Papatya building
Residents of Papatya building

Figure 46 | GUI Groups